

Docket No. AUS000153US1

### CLAIMS:

What is claimed:

1. A method on a server in a distributed data processing system for maintaining a logical composite repository of Object Identifier (OID) tree structures, the method comprising the steps of:
- receiving, in an OID abstraction layer, an OID tree structure from a repository;
  - registering the OID tree structure with a registry associated with the OID abstraction layer; and
  - adding the OID tree structure to a repository associated with the OID abstraction layer.
2. The method of claim 1, wherein the registry associated with the OID abstraction layer provides information identifying an anchor point in the OID subtree structure to be maintained by the repository.
3. The method of claim 2, wherein if the anchor point of the OID subtree structure is already registered with the OID abstraction layer, the registry is overwritten.
4. The method of claim 2, wherein if a query is received for an object that has an Object Identifier that is below a registered anchor point in an OID tree structure, the OID abstraction layer identifies a repository that maintains object information for the requested object based on the registered anchor point.
5. The method of claim 1, wherein the repository is configured such that the repository recognizes requests

Docket No. AUS000153US1

from an application program interface (API) associated with the OID abstraction layer and sends reply messages to the API containing information retrieved from the repository.

5

6. The method of claim 5, wherein the OID abstraction layer receives the information retrieved from the repository through the API and encapsulates the information in a reply message to a protocol interface.

10

7. The method of claim 1, wherein the OID abstraction layer receives a request for object data from a protocol interface, converts the request into an application program interface (API) request which is forwarded to the repository, and receives an API reply from the repository having the object data.

15

8. The method of claim 7, wherein the OID abstraction layer reformats the object data in a reply message and sends the reply message to the protocol interface.

20

9. A method on a server in a distributed data processing system for retrieving object data from a repository, comprising:

25

receiving a first query for the object data from a requester in the distributed data processing system, wherein the first query is in a protocol recognized by an OID abstraction layer;

30

locating a repository that contains the object data requested in the first query based on a registry associated with the OID abstraction layer; and

0958344-053100

retrieving the object data from the repository using an OID abstraction layer application program interface (API).

- 5 10. The method of claim 9, wherein the first query is mapped into a second query, wherein the second query is consistent with an application program interface (API) associated with the OID abstraction layer.
- 10 11. The method of claim 10, wherein if the first query cannot be mapped into a second query due to a limitation of the repository that contains the object associated with the first query, then the first query cannot be satisfied.
- 15 12. The method of claim 10, wherein the second query is sent to the repository that contains the object associated with the first query.
- 20 13. The method of claim 12, wherein a first reply is received at the API associated with the OID abstraction layer from the repository that contains the object associated with the first query.
- 25 14. The method of claim 13, wherein the first reply is transformed into a second reply, wherein the second reply is consistent with the protocol for the first query recognized by the OID abstraction layer.
- 30 15. The method of claim 14, wherein the second reply is sent to the requester in the distributed data processing system.

095834.1-053100

16. The method of claim 9, wherein each repository in the plurality of repositories contains information representing an Object Identifier (OID) subtree structure.
17. The method of claim 9, wherein Simple Network Management Protocol (SNMP) is a protocol recognized by the OID abstraction layer.
18. The method of claim 9, wherein Lightweight Directory Access Protocol (LDAP) is a protocol recognized by the OID abstraction layer.
19. The method of claim 9, wherein Common Information Model used in conjunction with eXtensible Markup Language (CIM/XML) is a protocol recognized by the OID abstraction layer.
20. An apparatus on a server in a distributed data processing system for maintaining a logical composite repository of Object Identifier (OID) tree structures, the apparatus comprising:
- an OID abstraction layer that receives an OID tree structure from a repository;
  - a registry, associated with the OID abstraction layer, that registers the OID tree structure; and
  - an adding means for adding the OID tree structure to a repository associated with the OID abstraction layer.

the apparatus of claim 20, wherein the information identifying an anchor point of the tree structure to be maintained by the apparatus of claim 21, wherein the information of the OID tree structure is already in the registry, then the registry overwrites the information.

the apparatus of claim 21, wherein the application layer receives a query for an Object Identifier that is below a root node in an OID tree structure, the registry identifies a repository containing the information for the requested object, and the application layer returns the requested anchor point.

the apparatus of claim 20, wherein the application layer is configured such that the repository receives information from an application program interacting with the OID abstraction layer, and the application layer returns the information to the API containing information about the repository.

the apparatus of claim 24, wherein the application layer receives the information about the repositories through the API and encodes the information in a reply message to a protocol interface.

the apparatus of claim 20, wherein the application layer receives a request for information through a protocol interface, converts the request into a query for the repository, and returns the information to the protocol interface.

22. The apparatus of claim 21, wherein if the anchor point of the OID tree structure is already registered in the registry, then the registry overwrites the previous entry.

23. The apparatus of claim 21, wherein, if the OID abstraction layer receives a query for an object that has an Object Identifier that is below a registered anchor point in an OID tree structure, the registry in the OID abstraction layer identifies a repository that maintains object information for the requested object based on the registered anchor point.

24. The apparatus of claim 20, wherein the repository is configured such that the repository recognizes requests received from an application program interface (API) associated with the OID abstraction layer and sends reply messages to the API containing information retrieved from the repository.

25 25. The apparatus of claim 24, wherein the OID  
abstraction layer receives the information retrieved from  
the repositories through the API and encapsulates the  
information in a reply message to a protocol interface.

30 26. The apparatus of claim 20, wherein the OID  
abstraction layer receives a request for object data from  
a protocol interface, converts the request into an

Docket No. AUS000153US1

application program interface (API) request which is forwarded to the repository, and receives an API reply from the repository having the object data.

5 27. The apparatus of claim 26, wherein the OID abstraction layer encapsulates the object data in a reply message and sends the reply message to the protocol interface.

10 28. An apparatus on a server in a distributed data processing system for retrieving object data from a repository, comprising:

15 a receiving means for receiving a first query for the object data from a requester in the distributed data processing system, wherein the first query is in a protocol recognized by an OID abstraction layer;

20 a locating means for locating a repository that contains the object data requested in the first query based on a registry associated with the OID abstraction layer; and

a retrieving means for retrieving the object data from the repository using an OID abstraction layer application program interface (API).

25 29. The apparatus of claim 28, further comprising a mapping means for mapping the first query into a second query, wherein the second query is consistent with an application program interface (API) associated with the OID abstraction layer.

30 30. The apparatus of claim 29, wherein if the mapping means cannot map the first query into a second query due

055834.1-053100

mitation of the repository that contains the object associated with the first query, then the request cannot be satisfied.

The apparatus of claim 29, further comprising sending means, in the OID abstraction, for sending the second query to a repository that contains the object associated with the first query.

The apparatus of claim 31, wherein the requester receives a first reply at the API from the repository that contains the object associated with the first query.

The apparatus of claim 32, further comprising transforming means, in the OID abstraction, for transforming the first reply into a second reply, wherein the second reply is consistent with the protocol and the query recognized by the OID abstraction.

The apparatus of claim 33, further comprising sending means, in the OID abstraction, for sending the second reply to the requester in the distributed data processing system.

The apparatus of claim 28, wherein each node contains an Object Identifier (OID) tree structure.

The apparatus of claim 28, wherein the requester recognizes a Simple Network Management Protocol (SNMP) query.

- 5 31. The apparatus of claim 29, further comprising a first sending means, in the OID abstraction layer, that sends the second query to a repository that contains the object associated with the first query.
- 10 32. The apparatus of claim 31, wherein the retrieving means receives a first reply at the API from the repository that contains the object associated with the first query.
- 15 33. The apparatus of claim 32, further comprising a transforming means, in the OID abstraction layer, that transforms the first reply into a second reply, wherein the second reply is consistent with the protocol for the first query recognized by the OID abstraction layer.
- 20 34. The apparatus of claim 33, further comprising a second sending means, in the OID abstraction layer, that sends the second reply to the requester in the distributed data processing system.
- 25 35. The apparatus of claim 28, wherein each repository contains Object Identifier (OID) tree structures.
- 30 36. The apparatus of claim 28, wherein the receiving means recognizes a Simple Network Management Protocol (SNMP) query.

Docket No. AUS000153US1

37. The apparatus of claim 28, wherein the receiving means recognizes a Lightweight Directory Access Protocol (LDAP) query.

5 38. The apparatus of claim 28, wherein the receiving means recognizes a Common Information Model used in conjunction with eXtensible Markup Language (CIM/XML) query.

10 39. A computer program product in a computer readable medium for maintaining a repository of Object Identifier (OID) tree structures, comprising:

instructions for receiving, in an OID abstraction layer, an OID tree structure from a repository;

15 instructions for registering the OID tree structure with a registry associated with the OID abstraction layer; and

instructions for adding the OID tree structure to a repository associated with the OID abstraction layer.

20

40. The computer program product of claim 39, further comprising instructions for maintaining the registry associated with the OID abstraction layer and providing information identifying an anchor point in the OID tree structure to be maintained by the repository.

25

41. The computer program product of claim 40, wherein if the anchor point of the OID tree structure is already registered with the OID abstraction layer, the instructions for registering overwrites the previous entry.

30

095834.1.053100

Docket No. AUS000153US1

42. The computer program product of claim 40, further comprising instructions for identifying a repository that maintains object information for the requested object based on the registered anchor point if a query is  
5 received for an object that has an Object Identifier that is below a registered anchor point in an OID tree structure.

43. The computer program product of claim 39, further  
10 comprising instructions for configuring the repository to recognize requests from an application program interface (API) associated with the OID abstraction layer and to send reply messages to the API containing information retrieved from the repository.

44. The computer program product of claim 43, further comprising instructions for receiving the information retrieved from the repository, through the API, and encapsulating the information in a reply message to a  
15  
20 protocol interface.

45. The computer program product of claim 39, further comprising instructions for receiving a request for object data from a protocol interface;

25 instructions for converting the request into an application program interface (API) request which is forwarded to the subtree repository; and

instructions for receiving an API reply from the subtree repository having the object data.

30 46. The computer program product of claim 45, further comprising instructions for encapsulating the object data

090341.053100

Docket No. AUS000153US1

in a reply message and sending the reply message to the protocol interface.

47. A computer program product in a computer readable  
5 medium for retrieving object data from a repository,  
comprising:

instructions for receiving a first query for the  
object data from a requester in the distributed data  
processing system, wherein the first query is in a  
10 protocol recognized by an OID abstraction layer;  
instructions for locating a repository that contains  
the object data requested in the first query based on a  
registry associated with the OID abstraction layer; and  
instructions for retrieving the object data from the  
15 repository using an OID abstraction layer application  
program interface (API).

48. The computer program product of claim 47, wherein  
the instructions for receiving the first query map the  
20 first query into a second query, wherein the second query  
is consistent with an application program interface (API)  
associated with the OID abstraction layer.

49. The computer program product of claim 48, wherein if  
25 the instructions for receiving the first query map cannot  
map the first query into a second query due to a  
limitation of the repository that contains the object  
associated with the first query, then the first query  
cannot be satisfied.

30

50. The computer program product of claim 48, further  
comprising instructions for sending the second query to

00523411-053100



The repository that contains the object of the first query.

The computer program product of claim 1, comprising instructions for receiving a first query from an API associated with the OID abstraction layer, the repository that contains the object of the first query.

The computer program product of claim 1, comprising instructions for transforming the first query to a second reply, wherein the second reply is consistent with the protocol for the first query as recognized by the OID abstraction layer.

The computer program product of claim 1, comprising instructions for sending the second reply to the requester in the distributed data structure.

The computer program product of claim 1, comprising instructions for receiving a first query from a Simple Network Management Protocol (SNMP) agent.

The computer program product of claim 1, comprising instructions for receiving a first query from a Lightweight Directory Access Protocol (LDAP) client.

5

10

15

20

25

30

Docket No. AUS000153US1

57. The computer program product of claim 47, wherein instructions for receiving a first query recognize a Common Information Model used in conjunction with eXtensible Markup Language (CIM/XML) query.

5

001E50"TF4E8560